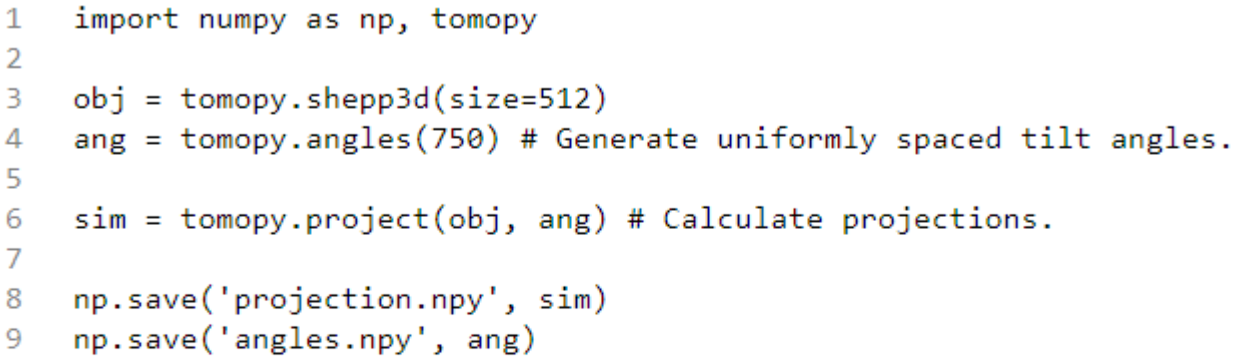
OPTIMIZING TOMOPY

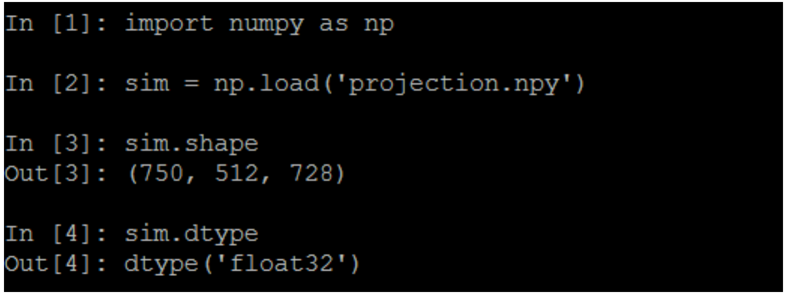
Performance analysis of grid reconstruction

Benchmarking: make_data.py



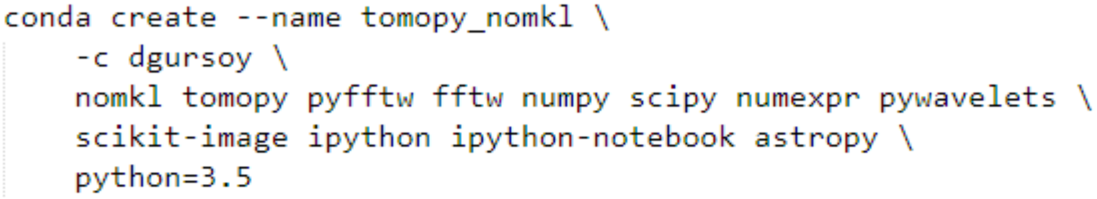
```python
import numpy as np, tomopy

obj = tomopy.shepp3d(size=512)
ang = tomopy.angles(750) # Generate uniformly spaced tilt angles.

sim = tomopy.project(obj, ang) # Calculate projections.

np.save('projection.npy', sim)
np.save('angles.npy', ang)
```

Checking dimensions and type of the projection data:

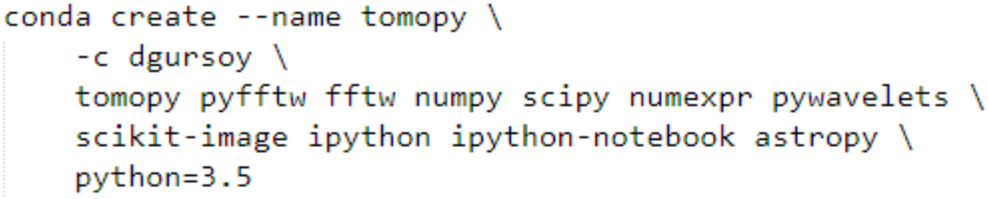


```
In [1]: import numpy as np

In [2]: sim = np.load('projection.npy')

In [3]: sim.shape
Out[3]: (750, 512, 728)

In [4]: sim.dtype
Out[4]: dtype('float32')
```

Tomopy out of the box

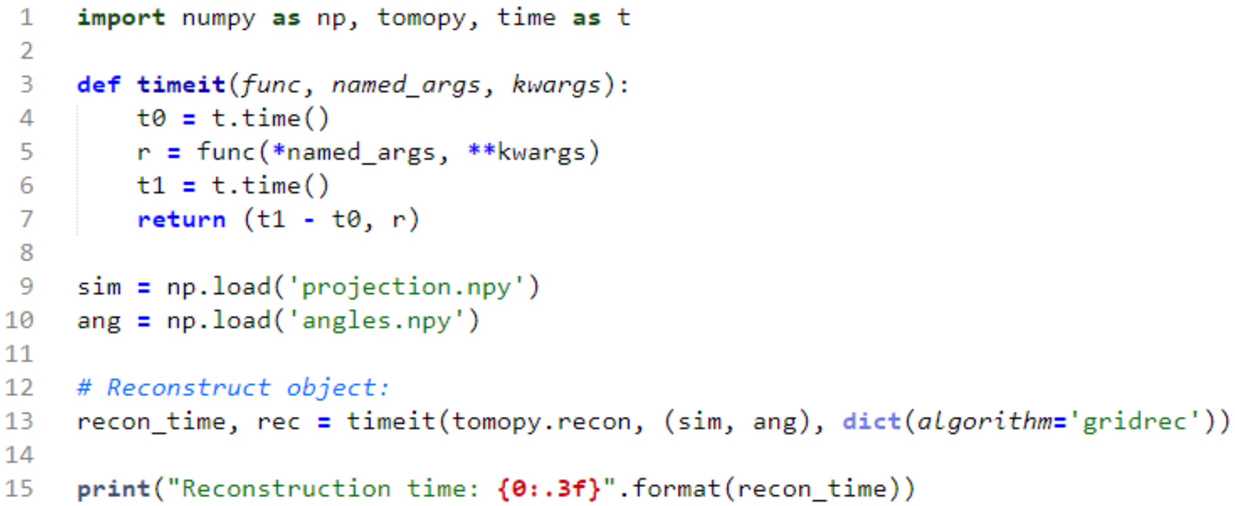


```
conda create --name tomopy_nomkl \
    -c dgursoy \
    nomkl tomopy pyfftw fftw numpy scipy numexpr pywavelets \
    scikit-image ipython ipython-notebook astropy \
    python=3.5
```



```
conda create --name tomopy \
    -c dgursoy \
    tomopy pyfftw fftw numpy scipy numexpr pywavelets \
    scikit-image ipython ipython-notebook astropy \
    python=3.5
```

Reconstruction script

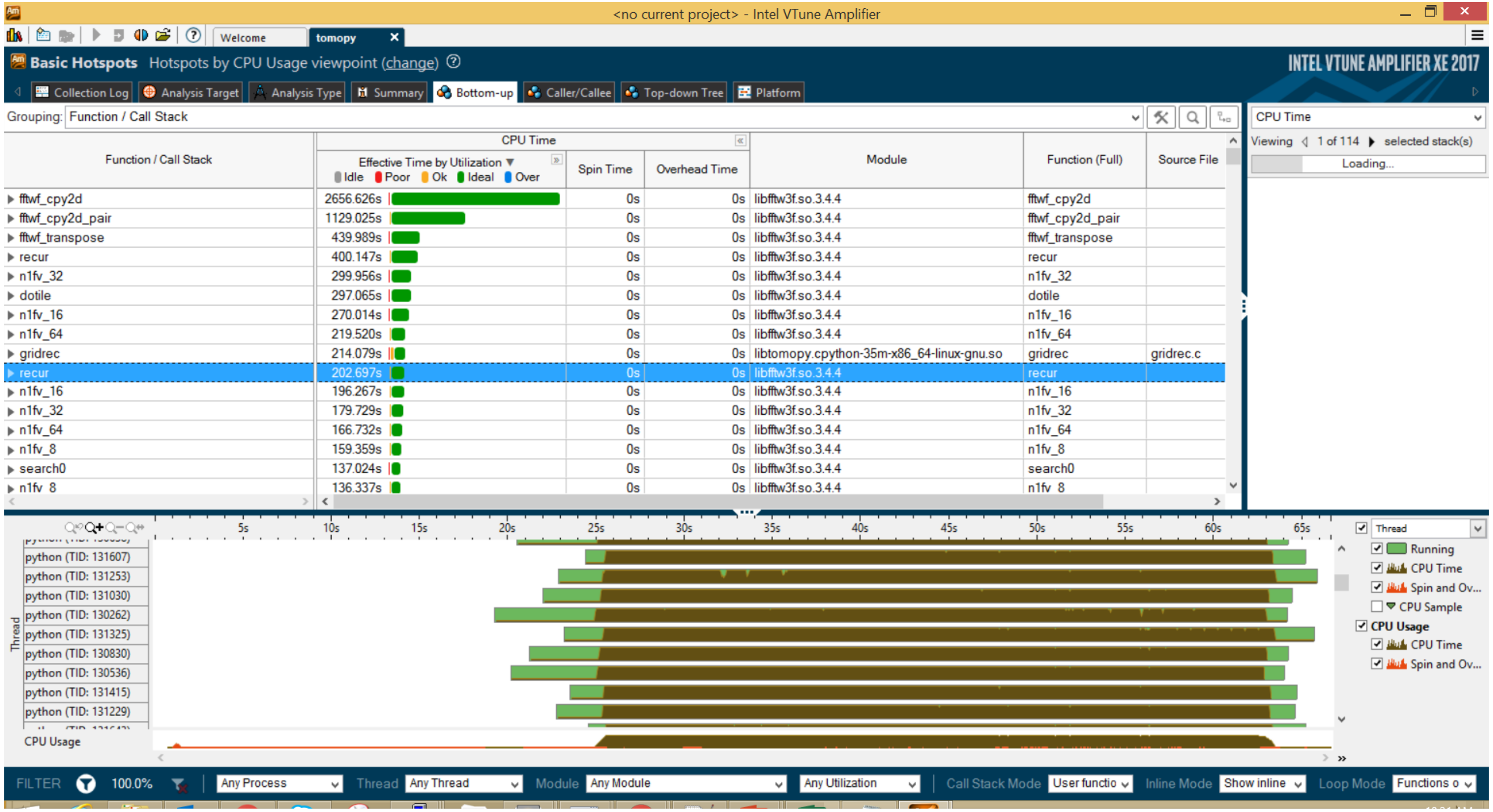


```python
import numpy as np, tomopy, time as t

def timeit(func, named_args, kwargs):
    t0 = t.time()
    r = func(*named_args, **kwargs)
    t1 = t.time()
    return (t1 - t0, r)

sim = np.load('projection.npy')
ang = np.load('angles.npy')

# Reconstruct object:
recon_time, rec = timeit(tomopy.recon, (sim, ang), dict(algorithm='gridrec'))

print("Reconstruction time: {0:.3f}".format(recon_time))
```

Performance times

(knl)$ numactl -p 1 \ python recon_bench.py					
KNL	nomkl	256	47.696		
KNL	mkl	256	98.56		must set OMP_NUM_THREADS=1
KNL	mkl	256	12.965		KMP_AFFINITY=disabled
(hsw)$ python recon_bench.py					
HSW	nomkl	32	4.246		
HSW	mkl	32	11.356		
HSW	mkl_seq	32	3.294		MKL_THREADING_LAYER=SEQUENTIAL

Hotspots tomopy_nomkl on KNL



<no current project> - Intel VTune Amplifier

Welcome | tomopy

Basic Hotspots Hotspots by CPU Usage viewpoint (change)

INTEL VTUNE AMPLIFIER XE 2017

Collection Log | Analysis Target | Analysis Type | Summary | Bottom-up | Caller/Callee | Top-down Tree | Platform

Grouping: Function / Call Stack

Function / Call Stack	CPU Time: Effective Time by Utilization	CPU Time: Spin Time	CPU Time: Overhead Time	Module	Function (Full)	Source File
fftwf_cpy2d	2656.626s	0s	0s	libfftw3f.so.3.4.4	fftwf_cpy2d	
fftwf_cpy2d_pair	1129.025s	0s	0s	libfftw3f.so.3.4.4	fftwf_cpy2d_pair	
fftwf_transpose	439.989s	0s	0s	libfftw3f.so.3.4.4	fftwf_transpose	
recur	400.147s	0s	0s	libfftw3f.so.3.4.4	recur	
n1fv_32	299.956s	0s	0s	libfftw3f.so.3.4.4	n1fv_32	
dotile	297.065s	0s	0s	libfftw3f.so.3.4.4	dotile	
n1fv_16	270.014s	0s	0s	libfftw3f.so.3.4.4	n1fv_16	
n1fv_64	219.520s	0s	0s	libfftw3f.so.3.4.4	n1fv_64	
gridrec	214.079s	0s	0s	libtomopy.cpython-35m-x86_64-linux-gnu.so	gridrec	gridrec.c
recur	202.697s	0s	0s	libfftw3f.so.3.4.4	recur	
n1fv_16	196.267s	0s	0s	libfftw3f.so.3.4.4	n1fv_16	
n1fv_32	179.729s	0s	0s	libfftw3f.so.3.4.4	n1fv_32	
n1fv_64	166.732s	0s	0s	libfftw3f.so.3.4.4	n1fv_64	
n1fv_8	159.359s	0s	0s	libfftw3f.so.3.4.4	n1fv_8	
search0	137.024s	0s	0s	libfftw3f.so.3.4.4	search0	
n1fv_8	136.337s	0s	0s	libfftw3f.so.3.4.4	n1fv_8	

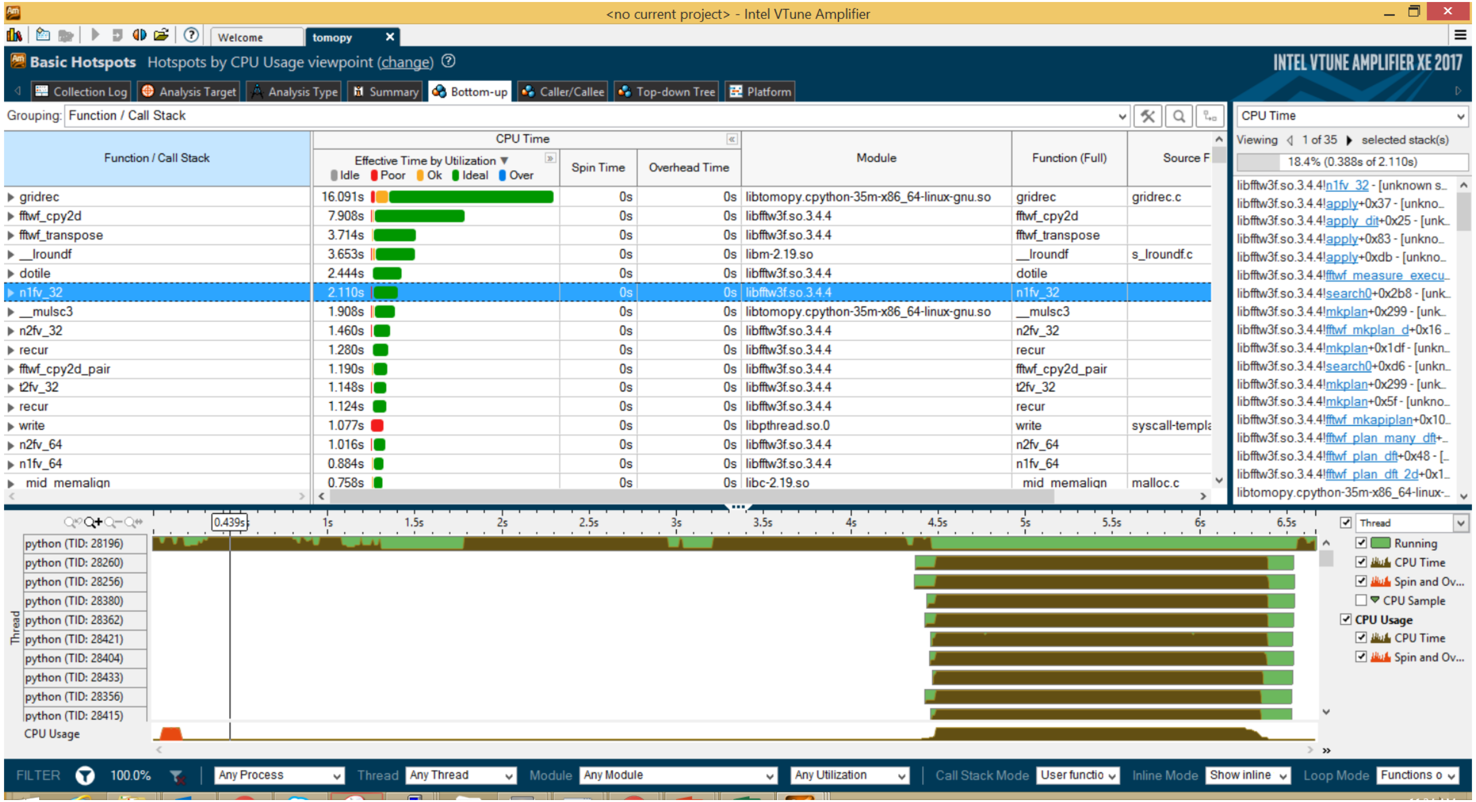
Idle | Poor | Ok | Ideal | Over

CPU Time — Viewing 1 of 114 selected stack(s) — Loading...

FILTER 100.0% | Any Process | Thread: Any Thread | Module: Any Module | Any Utilization | Call Stack Mode: User functio | Inline Mode: Show inline | Loop Mode: Functions o

Optimization Notice

Hotspots tomopy_nomkl on HSW



<no current project> - Intel VTune Amplifier

Welcome | tomopy

Basic Hotspots Hotspots by CPU Usage viewpoint (change)

INTEL VTUNE AMPLIFIER XE 2017

Collection Log | Analysis Target | Analysis Type | Summary | Bottom-up | Caller/Callee | Top-down Tree | Platform

Grouping: Function / Call Stack

Function / Call Stack	CPU Time: Effective Time by Utilization (▮ Idle ▮ Poor ▮ Ok ▮ Ideal ▮ Over)	Spin Time	Overhead Time	Module	Function (Full)	Source F
gridrec	16.091s	0s	0s	libtomopy.cpython-35m-x86_64-linux-gnu.so	gridrec	gridrec.c
fftwf_cpy2d	7.908s	0s	0s	libfftw3f.so.3.4.4	fftwf_cpy2d	
fftwf_transpose	3.714s	0s	0s	libfftw3f.so.3.4.4	fftwf_transpose	
__lroundf	3.653s	0s	0s	libm-2.19.so	__lroundf	s_lroundf.c
dotile	2.444s	0s	0s	libfftw3f.so.3.4.4	dotile	
n1fv_32	2.110s	0s	0s	libfftw3f.so.3.4.4	n1fv_32	
__mulsc3	1.908s	0s	0s	libtomopy.cpython-35m-x86_64-linux-gnu.so	__mulsc3	
n2fv_32	1.460s	0s	0s	libfftw3f.so.3.4.4	n2fv_32	
recur	1.280s	0s	0s	libfftw3f.so.3.4.4	recur	
fftwf_cpy2d_pair	1.190s	0s	0s	libfftw3f.so.3.4.4	fftwf_cpy2d_pair	
t2fv_32	1.148s	0s	0s	libfftw3f.so.3.4.4	t2fv_32	
recur	1.124s	0s	0s	libfftw3f.so.3.4.4	recur	
write	1.077s	0s	0s	libpthread.so.0	write	syscall-templa
n2fv_64	1.016s	0s	0s	libfftw3f.so.3.4.4	n2fv_64	
n1fv_64	0.884s	0s	0s	libfftw3f.so.3.4.4	n1fv_64	
mid_memalign	0.758s	0s	0s	libc-2.19.so	mid_memalign	malloc.c

CPU Time

Viewing 1 of 35 selected stack(s)

18.4% (0.388s of 2.110s)

```
libfftw3f.so.3.4.4!n1fv_32 - [unknown s...
libfftw3f.so.3.4.4!apply+0x37 - [unkno...
libfftw3f.so.3.4.4!apply_dit+0x25 - [unk...
libfftw3f.so.3.4.4!apply+0x83 - [unkno...
libfftw3f.so.3.4.4!apply+0xdb - [unkno...
libfftw3f.so.3.4.4!fftwf_measure_execu...
libfftw3f.so.3.4.4!search0+0x2b8 - [unk...
libfftw3f.so.3.4.4!mkplan+0x299 - [unk...
libfftw3f.so.3.4.4!fftwf_mkplan_d+0x16 ...
libfftw3f.so.3.4.4!mkplan+0x1df - [unkn...
libfftw3f.so.3.4.4!search0+0xd6 - [unkn...
libfftw3f.so.3.4.4!mkplan+0x299 - [unk...
libfftw3f.so.3.4.4!mkplan+0x5f - [unkno...
libfftw3f.so.3.4.4!fftwf_mkapiplan+0x10...
libfftw3f.so.3.4.4!fftwf_plan_many_dft+...
libfftw3f.so.3.4.4!fftwf_plan_dft+0x48 - [...
libfftw3f.so.3.4.4!fftwf_plan_dft_2d+0x1...
libtomopy.cpython-35m-x86_64-linux-...
```

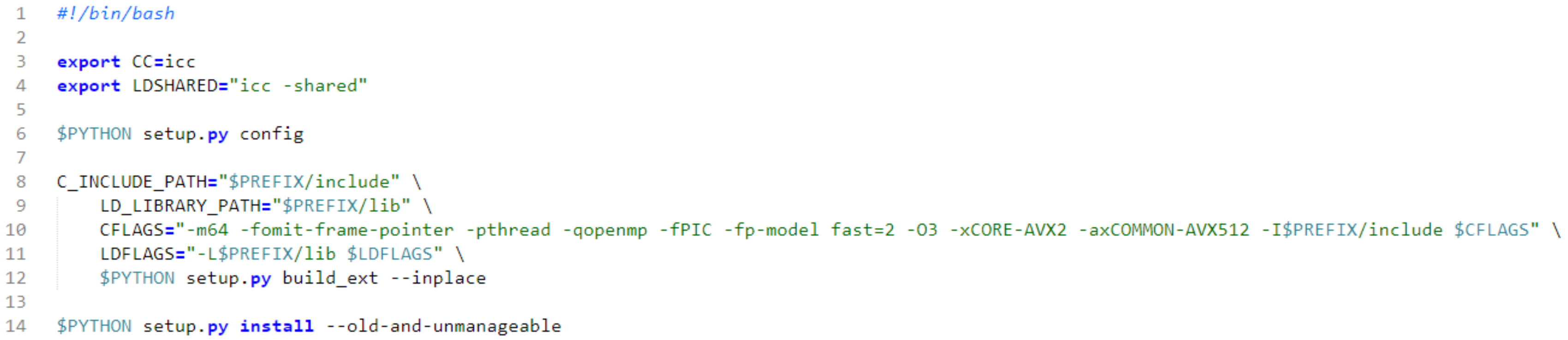
FILTER 100.0% | Any Process | Thread: Any Thread | Module: Any Module | Any Utilization | Call Stack Mode: User functio | Inline Mode: Show inline | Loop Mode: Functions o

Optimization Notice

Building tomopy toolchain with icc

- Created recipes to build essential components with icc targeting common-avx512 architechture
- Small changes to tomopy itself
  - removed –lm in setup.py, vectorized code in phantom.py
  - changes to gridrec.c to enable vectorization
- Modules are fftw, pyfftw, tomopy, dxchange, dxfile, olefile are built locally
- Modules numpy, scipy, scikit-image are conda-installed from intel channel
- Other modules (pywavelets, etc) taken from dgursoy channel
- netCDF4 and atropy were pip or conda installed

Tomopy recipe: build.sh



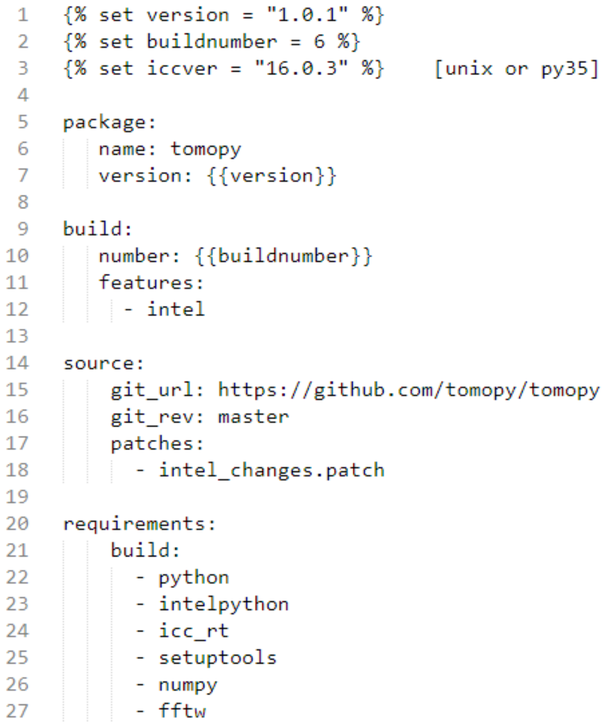
```
#!/bin/bash

export CC=icc
export LDSHARED="icc -shared"

$PYTHON setup.py config

C_INCLUDE_PATH="$PREFIX/include" \
    LD_LIBRARY_PATH="$PREFIX/lib" \
    CFLAGS="-m64 -fomit-frame-pointer -pthread -qopenmp -fPIC -fp-model fast=2 -O3 -xCORE-AVX2 -axCOMMON-AVX512 -I$PREFIX/include $CFLAGS" \
    LDFLAGS="-L$PREFIX/lib $LDFLAGS" \
    $PYTHON setup.py build_ext --inplace

$PYTHON setup.py install --old-and-unmanageable
```

Compile tomopy using icc targeting both HSW and KNL, enabling vectorization.

Recipes are available on cori.

Used vectorization report (–qopt-report=5) to guide optimizations

Tomopy recipe, cont.



```
{% set version = "1.0.1" %}
{% set buildnumber = 6 %}
{% set iccver = "16.0.3" %}    [unix or py35]

package:
   name: tomopy
   version: {{version}}

build:
   number: {{buildnumber}}
   features:
     - intel

source:
    git_url: https://github.com/tomopy/tomopy
    git_rev: master
    patches:
      - intel_changes.patch

requirements:
    build:
      - python
      - intelpython
      - icc_rt
      - setuptools
      - numpy
      - fftw
```

Patch represents diff between official

github.com/tomopy/tomopy.git

and branch **feature/intelem** of its fork

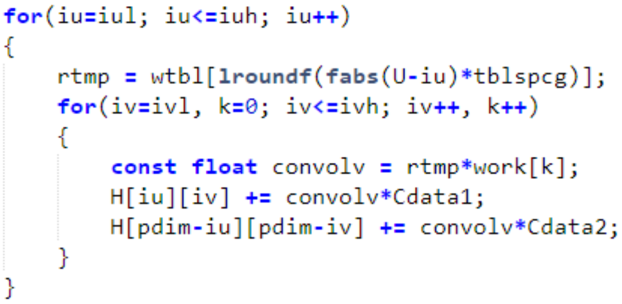
github.com/oleksandr-pavlyk/tomopy.git

Gist of optimizations

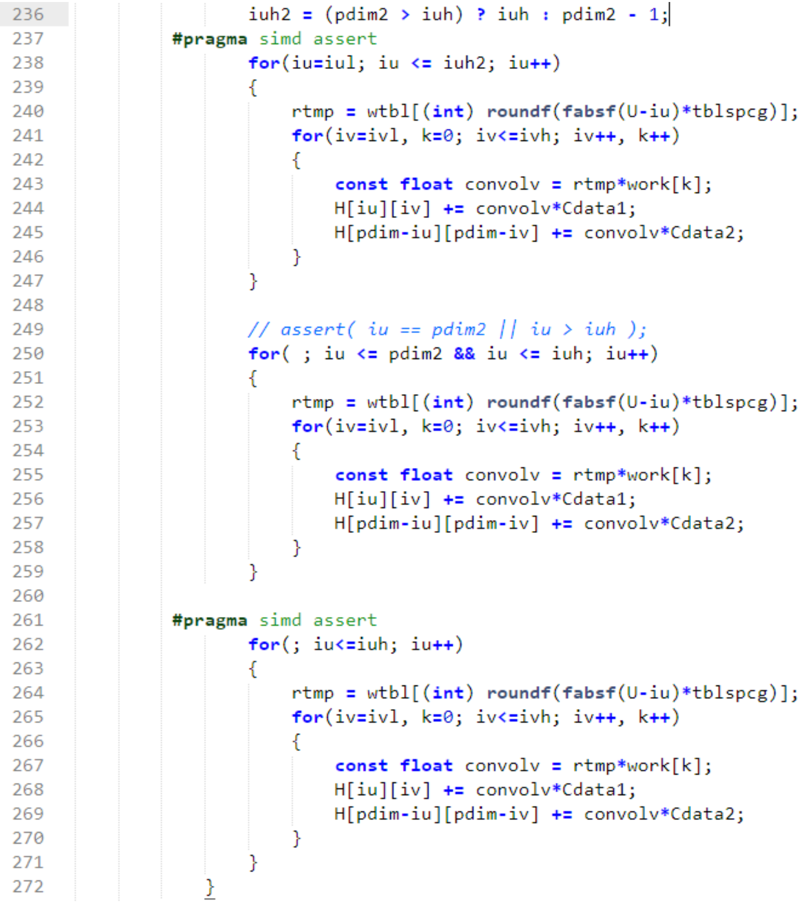
- Replace lroundf(x) with (int) roundf(x)
- Replace ceil(x) with ceilf(x), etc.
- Replace fabs(x) with fabs(f)
- Apply vectorization pragmas
- Split one double loop to enable vectorization

Optimization Notice

Changes in gridrec.c

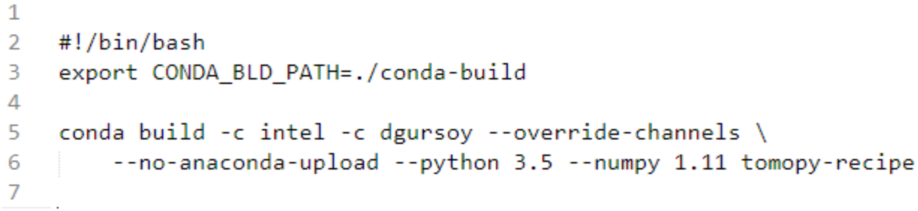


```
for(iu=iul; iu<=iuh; iu++)
{
    rtmp = wtbl[lroundf(fabs(U-iu)*tblspcg)];
    for(iv=ivl, k=0; iv<=ivh; iv++, k++)
    {
        const float convolv = rtmp*work[k];
        H[iu][iv] += convolv*Cdata1;
        H[pdim-iu][pdim-iv] += convolv*Cdata2;
    }
}
```



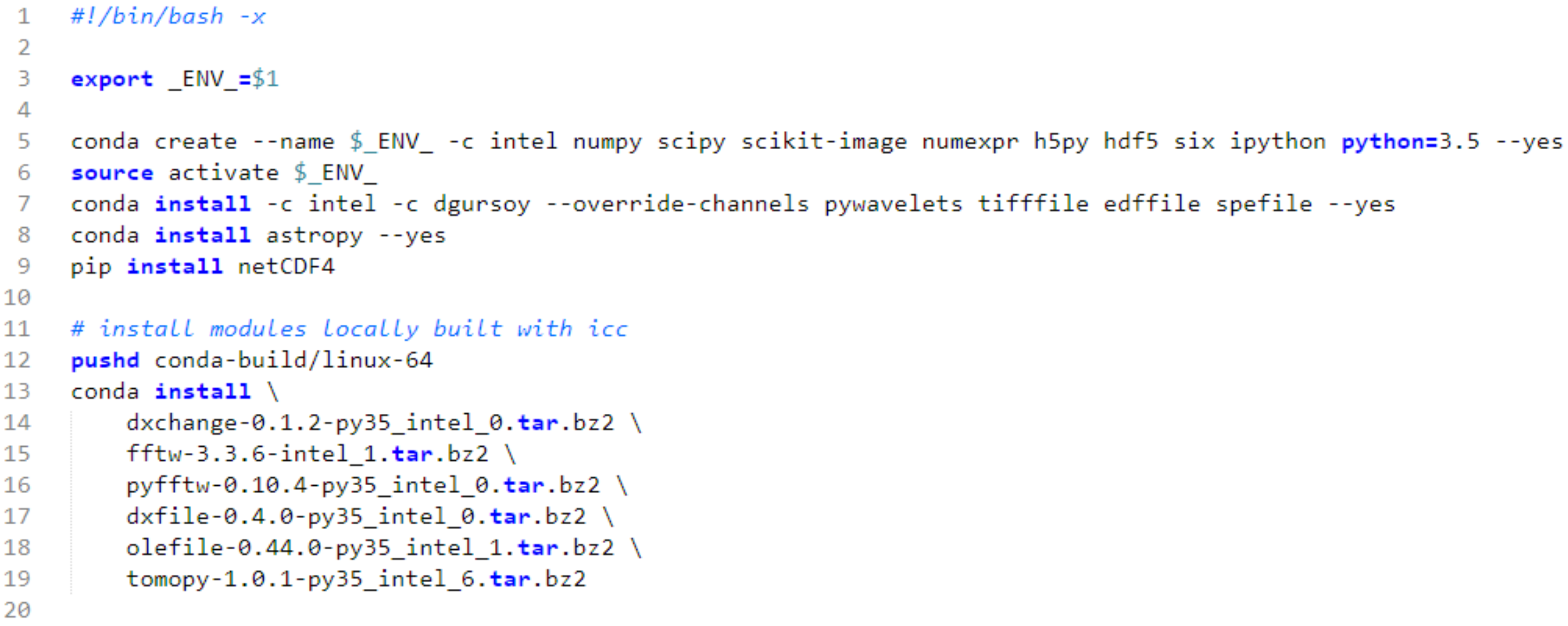
```
        iuh2 = (pdim2 > iuh) ? iuh : pdim2 - 1;
#pragma simd assert
        for(iu=iul; iu <= iuh2; iu++)
        {
            rtmp = wtbl[(int) roundf(fabsf(U-iu)*tblspcg)];
            for(iv=ivl, k=0; iv<=ivh; iv++, k++)
            {
                const float convolv = rtmp*work[k];
                H[iu][iv] += convolv*Cdata1;
                H[pdim-iu][pdim-iv] += convolv*Cdata2;
            }
        }

        // assert( iu == pdim2 || iu > iuh );
        for( ; iu <= pdim2 && iu <= iuh; iu++)
        {
            rtmp = wtbl[(int) roundf(fabsf(U-iu)*tblspcg)];
            for(iv=ivl, k=0; iv<=ivh; iv++, k++)
            {
                const float convolv = rtmp*work[k];
                H[iu][iv] += convolv*Cdata1;
                H[pdim-iu][pdim-iv] += convolv*Cdata2;
            }
        }

#pragma simd assert
        for(; iu<=iuh; iu++)
        {
            rtmp = wtbl[(int) roundf(fabsf(U-iu)*tblspcg)];
            for(iv=ivl, k=0; iv<=ivh; iv++, k++)
            {
                const float convolv = rtmp*work[k];
                H[iu][iv] += convolv*Cdata1;
                H[pdim-iu][pdim-iv] += convolv*Cdata2;
            }
        }
    }
```

Building tomopy-recipe



```bash
#!/bin/bash
export CONDA_BLD_PATH=./conda-build

conda build -c intel -c dgursoy --override-channels \
    --no-anaconda-upload --python 3.5 --numpy 1.11 tomopy-recipe
```

Using built tomopy



```
#!/bin/bash -x

export _ENV_=$1

conda create --name $_ENV_ -c intel numpy scipy scikit-image numexpr h5py hdf5 six ipython python=3.5 --yes
source activate $_ENV_
conda install -c intel -c dgursoy --override-channels pywavelets tifffile edffile spefile --yes
conda install astropy --yes
pip install netCDF4

# install modules locally built with icc
pushd conda-build/linux-64
conda install \
    dxchange-0.1.2-py35_intel_0.tar.bz2 \
    fftw-3.3.6-intel_1.tar.bz2 \
    pyfftw-0.10.4-py35_intel_0.tar.bz2 \
    dxfile-0.4.0-py35_intel_0.tar.bz2 \
    olefile-0.44.0-py35_intel_1.tar.bz2 \
    tomopy-1.0.1-py35_intel_6.tar.bz2

```

Performance results

python recon_bench.py										
HSW	optimized	32	1.343							
KNL	optimized	256	2.492		KMP_AFFINITY=disabled numactl -p 1 python recon_bench.py					

Optimization Notice

intel®

experience
what's inside™

Legal Disclaimer & Optimization Notice

INFORMATION IN THIS DOCUMENT IS PROVIDED "AS IS". NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

Copyright © 2015, Intel Corporation. All rights reserved. Intel, Pentium, Xeon, Xeon Phi, Core, VTune, Cilk, and the Intel logo are trademarks of Intel Corporation in the U.S. and other countries.

Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804